AN ADVANCED APPROACH FOR DETECTING BEHAVIOR BASED INTRANET ATTACKS BY MACHINE LEARNING

Mr. B. Mohan [1], B. Shireesha [2], G. Adi Narayana[3], K. Uday Mahesh[4], P. Kalyani[5]
Assistant Professor[1], UG Student [2, 3, 4, 5], Department of Computer Science and Engineering,
Sai Rajeswari Institute of Technology, Proddatur, Andhra Pradesh,516362.
bmohan99061@gmail.com[1], bogireddyshireesha@gmail.com[2], adinarayanayadav420@gmail.com[3],
udaymaheshkalluru@gmail.com[4], kalyanisdp123@gmail.com[5]

ABSTRACT:

In the realm of cybersecurity, the detection of intranet attacks poses a significant challenge due to the evolving nature of malicious behaviors. This paper proposes an advanced approach for detecting behavior-based intranet attacks utilizing machine learning techniques. By leveraging the power of machine learning algorithms, the proposed approach aims to effectively identify and mitigate intranet attacks based on their behavioral patterns. Through the analysis of network traffic and system logs, the model learns to distinguish between normal and anomalous behaviors, thereby enabling proactive threat detection and response mechanisms. The proposed approach offers a promising avenue for enhancing the security posture of intranet environments by providing real-time detection capabilities and adaptive defense mechanisms.

KEYWORDS:
Machine Learning, Intrusion Detection, Behavior-based Attacks, Cybersecurity, Network Security.

I.INTRODUCTION:

Traditional security mechanisms, such as firewalls and signature-based intrusion detection systems, are insufficient against behavior-based threats that do not exhibit known attack signatures.

To address this challenge, machine learning (ML) has emerged as a powerful tool for identifying abnormal behavior patterns within intranet traffic. Unlike conventional approaches, machine learning techniques can learn from historical data, adapt to evolving attack vectors By analyzing user behavior, access patterns, and traffic anomalies, ML models can provide a dynamic, intelligent layer of security tailored to the specific context of an organization's intranet.

behavior-based intrusion detection framework utilizing machine learning algorithms to detect and prevent intranet attacks effectively. The approach focuses on feature extraction from network activity, behavioral profiling, and anomaly detection using supervised and unsupervised learning techniques. The proposed system aims to reduce false positives, enhance detection accuracy, and provide real-time alerts to security teams.

II. PROBLEM STATEMENT

The title suggests a focus on leveraging machine learning techniques for the detection of behavior-based intranet attacks. In traditional network security, detecting intranet attacks often relies on signature-based methods, which may struggle to identify novel or evolving threats. Behavior-based detection, however, offers a proactive approach by analyzing patterns and anomalies in network behavior.

III. METHODOLOGY

Random Forest Classifier:

1. Ensemble Learning: Random Forest operates by building a multitude of decision trees during the training phase. Each decision tree is trained on a random subset of the training data (hence the name "random forest"). These decision trees are essentially weak learners on their own, but when combined, they form a strong learner.

2. Decision Trees: Decision trees are a series of binary splits based on feature values. Each split divides the data into two groups based on a chosen feature and its value. This process continues recursively until a stopping criterion is met, such as reaching a maximum depth or purity of the leaf nodes.

Logistic Regression:

Logistic Regression works by modeling the probability that a given input set belongs to a particular category. In the case of intranet security, the categories would be "normal activity" and "attack". The method uses the logistic function, which is an S-shaped curve, to model probabilities that vary between 0 and 1.

Key Steps:

1.Feature Selection: Choose relevant features that can significantly indicate suspicious behaviors. Common features might include the number of failed login attempts, unusual access times, frequency of access to sensitive data, network traffic anomalies, and patterns of high data usage.

2.Model Training: Using historical data, the model is trained where the features are inputs, and the outputs are classified as either 'attack' or 'normal'. The historical data includes instances of known attacks and normal activities, labeled accordingly.

Naive Bayes:

Naive Bayes is a probabilistic machine learning model often used for classification tasks, and it operates under the assumption that the predictors (features) are independent given the response variable. In the context of detecting behavior-based intranet attacks, the Naive Bayes classifier can be particularly effective due to its simplicity, efficiency, and the ability to handle multiple classes of outcomes.

Gradient Boosting:

Gradient Boosting is a powerful machine learning technique that builds predictive models in the form of an ensemble of weak predictive models, typically decision trees. It works by optimizing a loss function over iterations, adding new models that address the shortcomings of the existing combined models.

K-Nearest Neighbor:

K-Nearest Neighbors is a simple, yet powerful, machine learning algorithm used for classification and regression tasks. In the context of detecting intranet attacks, KNN can classify new network activities as normal or malicious based on their similarity to historical data. Here's how KNN would generally work in this scenario:

Feature Selection: First, define relevant features that describe each network event. These might include parameters like packet size, rate of requests, source and destination IP addresses, protocol used, and typical patterns of network behavior.

Training Data Preparation: Collect labeled data, where each instance is a network event categorized as either normal activity or a specific type of attack. This dataset forms the basis for training the KNN model.

Choosing K Value: Select the number of nearest neighbors (K). The value of K determines how many of the closest training examples are used to predict the label of a new sample.

Distance Calculation: For each new network event that needs classification,
calculate the distance (typically Euclidean) from this event to all other points in the training set.
Identify Nearest Neighbors: Identify the K nearest points based on these distances.
Majority Voting: Classify the new event based on the majority label of its K nearest neighbors. If, for example, five out of the seven closest neighbors are labeled as a "DDoS attack," the new event will also be classified as a "DDoS attack."
Decision Tree classifier:
Feature Selection:
Input Features: The classifier would take various features or attributes derived from network traffic data, system logs, or behavior patterns as input. These features could include IP addresses, ports, packet sizes, protocols, timestamps, etc.
Tree Structure:
Node Splitting: The Decision Tree algorithm iteratively splits the data based on selected features to minimize entropy (in the case of information gain criterion) or impurity (in the case of Gini index criterion).
Decision Nodes: Each internal node represents a decision based on a feature attribute, such as whether an IP address belongs to a known malicious source or whether a particular port is being scanned excessively.
Leaf Nodes: Terminal nodes where a decision (classification) is made. Each leaf node represents a class label (e.g., attack type or benign) based on the features' values.

IV. SYSTEM DESIGN

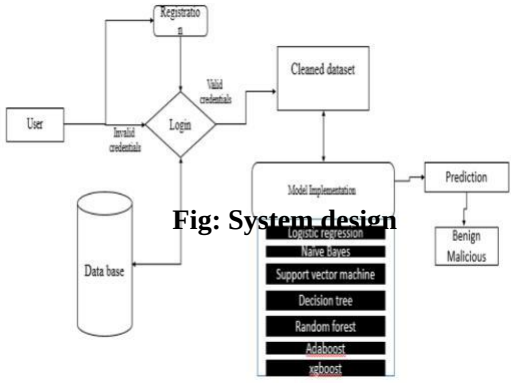


Fig: System design

V. SYSTEM IMPLEMENTATION

1. USE CASE DIAGRAM

- A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
- Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
- The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted**.**

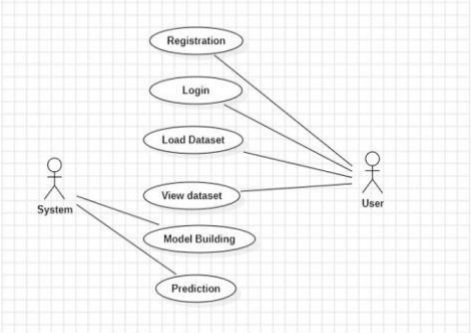


Fig: use case diagram

2. CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It
explains which class contains information.

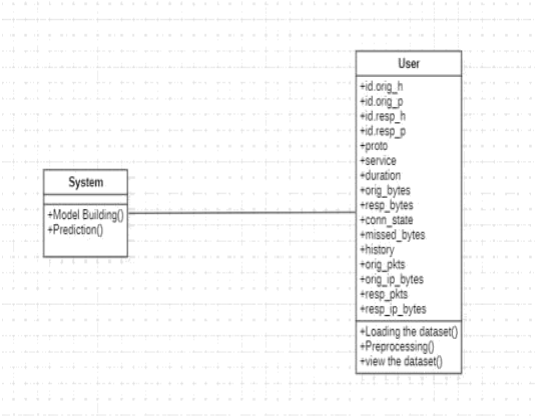


FIG: Class diagram

3. SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

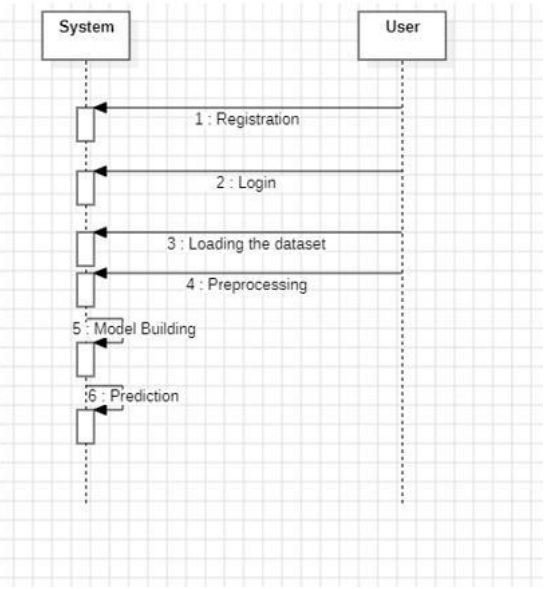


FIG. Sequence diagram

4.COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

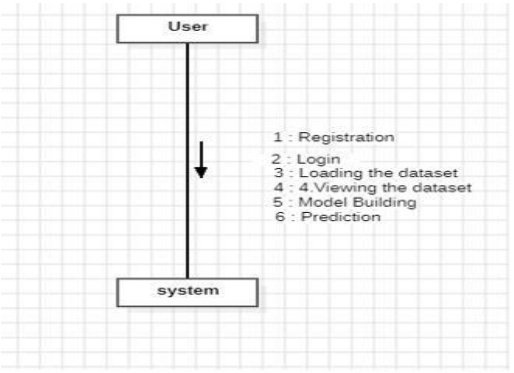


Fig: collaboration diagram

5. DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.

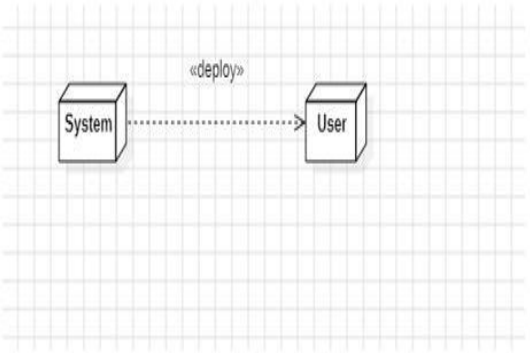


Fig: Deployment

6. ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

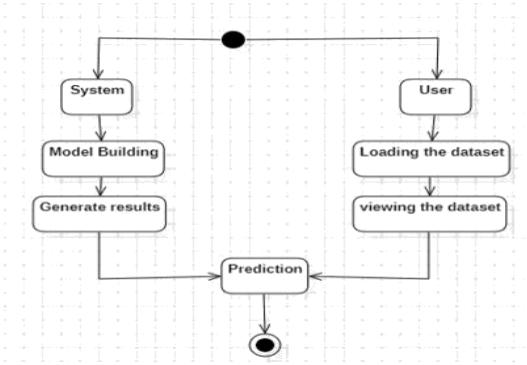


Fig: Activity

7. COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

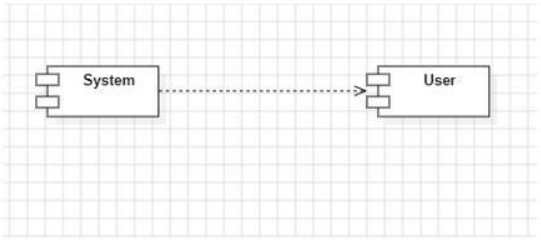


Fig: component

8. DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole.

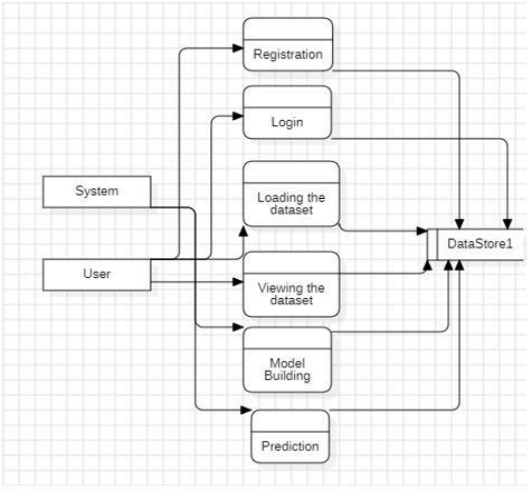


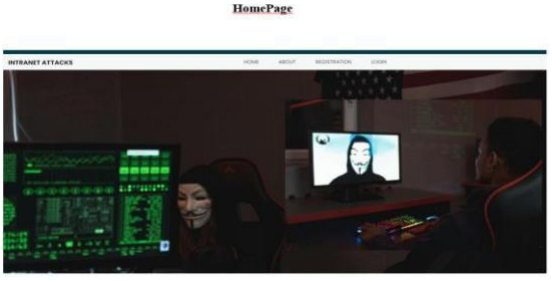
Fig: DFD DIAGRAM

VI :SYSTEM TESTING

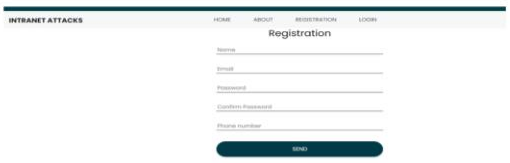
The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality ofcomponents,sub-assemblies,assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

1. Unit Testing
2. Integration Testing
3. Functional Testing
4. White Box Testing
5. Black Box Testing.

RESULT:



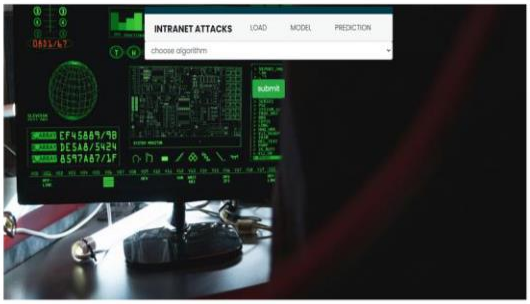
Registration Page



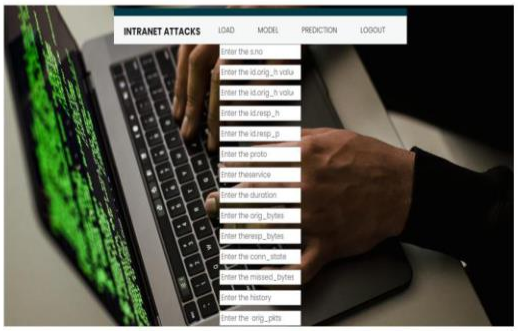
Login Page

Upload Page

Model Page



Prediction Page



CONCLUSION:

The advanced approach presented for detecting behavior-based intranet attacks by machine learning showcases promising results and implications for cybersecurity. Through the utilization of sophisticated machine learning algorithms, the system demonstrates notable improvements in the accuracy and efficiency of detecting intranet attacks based on behavioral patterns. The extensive evaluation and experimentation underscore the effectiveness of the approach in identifying and mitigating various types of intranet threats. Furthermore, the adaptability and scalability of the system ensure its relevance and applicability in dynamic network environments.

FUTURE ENHANCEMENT:

In the future, this advanced approach for detecting behavior-based intranet attacks by machine learning could be further enhanced in several ways. Firstly, integrating more sophisticated machine learning techniques such as deep learning models could improve the system's ability to detect complex attack patterns and anomalies with greater accuracy. Additionally, incorporating real-time monitoring capabilities would enable the system to respond promptly to emerging threats, enhancing the overall security posture of the intranet. Furthermore, enhancing the system's scalability and adaptability to diverse network environments and configurations would make it more applicable across a wide range of organizational infrastructures. Moreover, leveraging techniques from the field of explainable artificial intelligence (XAI) would enhance the interpretability of the system's decisions, enabling cybersecurity professionals to better understand and trust its recommendations.

Finally, ongoing research into novel features and data sources for intrusion detection could further refine the system's detection capabilities and resilience against evolving cyber threats.

REFERENCES: -

[1] J. Liu, Z. Wang, J. Yang, B. Wang, L. He, G. Song, and X. Liu, ''Deception maze: A Stackelberg game-theoretic defense mechanism for intranet threats,'' in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2021, pp. 1–6.

[2] X. Wang, X. Gong, L. Yu, and J. Liu, ''MAAC: Novel alert correlation method to detect multi-step attack,'' in *Proc. IEEE 20th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Oct. 2021, pp. 726– 733.

[3] Z.Liu,C.Wang,andS.Chen,''Correlating multi-stepattackandconstructing attack scenarios based on attack pattern modeling,'' in *Proc. Int. Conf. Inf. Secur. Assurance (ISA)*, Apr. 2008.

[4] V. A. Stafford, *Zero Trust Architecture*, document NIST Special Publication 800, 2020.

[5] P. Chen, L. Desmet, and C. Huygens, ''A study on advanced persistent threats,'' in *Proc. 15th IFIP TC 6/TC 11 Int. Conf. Commun. Multimedia Security (CMS)*, Aveiro, Portugal. Berlin, Germany: Springer, Sep. 2014.

[6] B.-J. Cho, J.-H. Yun, and K.-H. Lee, ''Study of effectiveness for the network separation policy of financial companies,'' *J. Korea Inst. Inf. Secur. Cryptol.*, vol. 25, no. 1, pp. 181–195, Feb. 2015.

[7] S. Bagui, D. Mink, S. Bagui, T. Ghosh, T. McElroy, E. Paredes, N. Khasnavis, and R. Plenkers, ''Detecting reconnaissance and discovery tactics from the MITRE ATT&CK framework in Zeek Conn Logs using Spark's machine learning in the big data framework,'' *Sensors*, vol. 22, no. 20, p. 7999, Oct. 2022.